Reduced Vlasov-Maxwell modeling

Philippe Helluy, Michel Massaro, Laurent Navoret, Nhung Pham, Thomas Strub

Reduced Vlasov-Maxwell modeling

P. Helluy[1], **M. Massaro**[2], **L. Navoret**[1], **N. Pham**[2], **and T. Strub**[3]

[1]Univ. Strasbourg & Inria Tonus, France
[2]Univ. Strasbourg, France
[3]AxesSim, France

Abstract— We describe CLAC (Conservation Laws Approximation on many Cores), a generic Discontinuous Galerkin (DG) solver for three-dimensional electromagnetic simulations. The solver runs on clusters of GPUs, it is based on hybrid parallelism using the OpenCL and MPI libraries. We explain how to solve the Vlasov-Maxwell equations with this tool. We present several numerical results.

1. INTRODUCTION

CLAC is a generic DG solver for conservation laws. The structure of the underlying hyperbolic system is described in an abstract manner. It can run on GPU clusters and it uses the MPI library for communications between GPUs in the cluster and OpenCL for driving the GPUs. The CLAC solver is very efficient for simulating industrial EM devices.

In plasma physics or beam physics applications, it is important to couple the Maxwell model with the Vlasov model, which describes the motions of charged particles. The unknown of the Vlasov equation is the distribution function of the charged particles, which depends on time and on a six-dimensional space-velocity variable.

In order to harness our multi-GPU solver we reduce the Vlasov equation to a space-only first order hyperbolic system compatible with the Maxwell model. We compute the coupled system with the generic DG solver.

We present two-dimensional and three-dimensional numerical simulations for the Maxwell or the Vlasov-Maxwell equations.

2. DG FOR CONSERVATION LAWS

2.1. Conservation laws

We consider a very general system of conservation laws

$$\partial_t W + \partial_i F^i(W) = S(W). \tag{1}$$

In this system, the unknown is a vector of conservative variables $W(x,t) \in \mathbb{R}^m$ that depends on the space variable $x = (x_1, x_2, x_3) \in \mathbb{R}^3$ and on the time t. We use the convention of sum on repeated indices. The source term is denoted by S. For a vector $n = (n_1, n_2, n_3) \in \mathbb{R}^3$ we shall also define the flux function by

$$F(W, n) = F^i(W) n_i. \tag{2}$$

The Maxwell equations enter this framework. In dimensionless form, They read

$$\partial_t E - \nabla_x \times H = -J, \tag{3}$$
$$\partial_t H + \nabla_x \times E = 0, \tag{4}$$

where E is the electric field, H the magnetic field and J the electric current. If we set

$$W = (E^T, H^T)^T, \quad S = (-J^T, (0,0,0)^T)^T,$$

and

$$F(W, n) = \begin{bmatrix} 0 & -n\times \\ n\times & 0 \end{bmatrix} W,$$

then the Maxwell equations are a particular case of (1), with $m = 6$.

2.2. General DG formalism

The DG approximation is defined on a mesh of the computational domain Ω. We consider a mesh made of a finite number of open cells. Let L and R be two neighbor cells. We denote by n_{LR} the unit normal vector on $\partial L \cap \partial R$ oriented from L to R.

In each cell L, we consider a basis of scalar function φ_i^L, $i = 1 \cdots p_L$ with support in L. It is possible to take different approximation order p_L on different cells. In cell L, the solution of (1) is approximated by

$$Wx,t) = W_L(x,t) = W_L^j(t)\varphi_j^L(x) \quad X \in L. \tag{5}$$

The numerical solution satisfies the DG approximation scheme (see for instance [1, 2, 3] and included references)

$$\forall L, \forall i \quad \int_L \partial_t W \varphi_i^L - \int_L F(W, \nabla_x \varphi_i^L) + \int_{\partial L} F(W_L, W_R, n_{LR})\varphi_i^L = \int_L S\varphi_i^L. \tag{6}$$

In practice the integrals on L or ∂L are perform numerically with Gauss-Legendre quadrature and the basis functions φ_i^L form an interpolation basis associated to the quadrature points in L.

Because W is discontinuous at the cell boundaries, it is not possible to define $F(W, n_{LR})$ on ∂L. Therefore, as in the finite volume method, we have to introduce a numerical flux $F(W_L, W_R, n_{LR})$. In practice, for stability reasons, we prefer an upwind numerical flux.

Introducing expansion (5) into (6) we obtain a system of ordinary differential equations satisfied by the basis coefficients $W_L^j(t)$. We solve the differential equations with a standard second order Runge-Kutta integrator.

3. REDUCED VLASOV-MAXWELL

In addition to the Maxwell equations, we can consider the motion of charged particles accelerated by the EM field. The distribution function $f(x, v, t)$ counts the particles at time t and position x having a velocity $v = (v^1, v^2, v^3)$. We denote by D the velocity domain, $v \in D$. The distribution function satisfies the Vlasov equation (written in dimensionless form)

$$\partial_t f + \nabla_x \cdot (fv) + \nabla_v \cdot (f(E + v \times H)) = 0. \tag{7}$$

The electric current J appearing in the Ampre equation is then given by

$$J = \int_{v \in \mathbb{R}^3} fv dv.$$

The difficulty is that the Vlasov equation (7) is set in a six-dimensional space. We propose a simple reduction procedure that allows to rewrite the Vlasov-Maxwell system under a system of conservations laws in the standard physical three-dimensional space. In this way we can apply our generic DG solver.

For this purpose, we consider a finite number of continuous basis functions depending on the velocity v

$$\varphi_i(v), \quad i = 1 \cdots P.$$

We assume that the velocity domain is large enough so that the basis functions vanish on the boundary of the velocity domain

$$v \in \partial D \Rightarrow \varphi_i(v) = 0.$$

We expand the distribution function in this basis

$$f(x, v, t) \simeq \sum_{j=1}^{P} f^j(x, t)\varphi_j(v) = f^j \varphi_j.$$

We insert this representation into the Vlasov equation (7), multiply by φ_i and integrate on the velocity domain D.

Defining the following $P \times P$ matrices

$$M_{i,j} = \int_v \varphi_i \varphi_j, \quad A_{i,j}^k = \int_v \varphi_i \varphi_j v^k, \quad B_{i,j} = -\int_v (E + v \times H) \cdot \nabla_v \varphi_i \varphi_j, \tag{8}$$

the Vlasov equation can be rewritten in the reduced form

$$M\partial_t w + A^k \partial_k w + Bw = 0, \quad w = (f^1, \cdots, f^P)^T. \tag{9}$$

The form (9) is called the reduced Vlasov equation. It is a first order hyperbolic system of conservation laws [4] that can also be solved by the standard DG solver. For practical reasons it is important to provide an efficient choice of basis functions φ_i. A good choice ensures a small number of basis functions P and that the matrices M, A^k and B are sparse. In practice, we use an adequate choice of numerical quadrature that leads to diagonal matrices M and A^k [5].

4. HYBRID OPENCL/MPI PARALLELISM

4.1. OpenCL parallelization

OpenCL library provides a model of parallel computing which can be used to drive GPUs and multicore accelerators. It is very similar to CUDA.

In this model, the memory is divided into global memory of some gigabytes with slow access and local cache memory of a few tens of kilobytes with quick access. Computations are performed by many "work-items" having access to all the global memory. The work-items are grouped into "work-groups" with common local memory.

All work-items execute the same program, called "kernel". Each work-item is identified by an index for parallelization.

The mesh is first split into several subdomains. Each subdomain is handled by one MPI node, associated to one GPU.

The OpenCL algorithm is divided into stages, each carried out by a specific kernel. These steps are (1) the calculation of the volume integral $\int_L F(W, \nabla\varphi_i^L)$ for each basis function φ_i^L, (2) the calculation of the surface integral $\int_{\partial L} F(W_L, W_R, n_{LR})\varphi_i^L$ for each basis function φ_i^L and (3) the Runge-Kutta step.

In the different OpenCL kernels participating to the element computations, the number of work-items depends on the degree of interpolation of the element.

A strategy for grouping elements of the same order into homogeneous zones is implemented. In this way the work-items of a same zone perform exactly the same operations, which improve the GPU efficiency.

4.2. MPI parallelization

MPI allows us to parallelize computations on distributed memory architecture by launching multiple process which communicate between each others. We perform a subdivision of the mesh into sub-domains. During the simulation, the boundary cells data are exchanged between sub-domains thanks to MPI send and receive operations. Before the MPI exchanges, the boundary data have also to be copied between the GPU and the CPU, which increases the simulation cost compared to other MPI parallel algorithms.

In order to avoid performance waste, we have implemented an asynchronous task graph in CLAC. This graph task allows to computing the interior of the zones while the boundaries are exchanged between the MPI nodes.

5. NUMERICAL RESULTS

In this section, we present the results obtained with three test cases.

5.1. EM simulation

The objective of this test case is to evaluate the MPI scalability of the computation.

We consider the mesh of a domain containing a generic airplane. The mesh is made of more than 3×10^6 hexaedrons with an second order approximation. This represents more than 10^9 degrees of freedom per time step. The airplane is illuminated by a plane wave with Gaussian profile polarized in E_y. The computational domain is surrounded by PML layers.

We use 8 NVIDIA K20 GPUs to perform the computation. The simulation does not fit into a single GPU memory. We obtain a time per iteration of 1.73s, which represents 362 Gflops. In this test case, if we deactivate the asynchronous graph task, we spend about 30% of the computation time in the memory transfers between the CPU and the GPU and about 20% in the MPI communications.

We plot on Figure 1 the surface current $|n \times H|$ on the body of the airplane at time 51 ns.

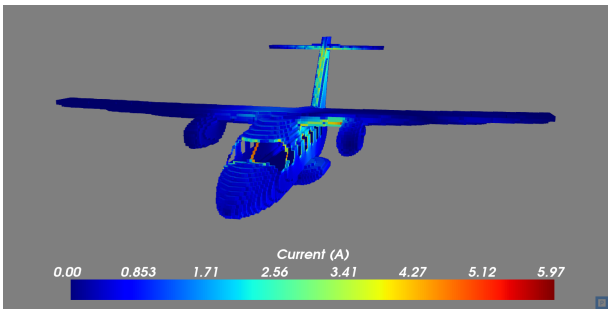

Figure 1: Generic airplane, current on the fuselage at time 51ns..

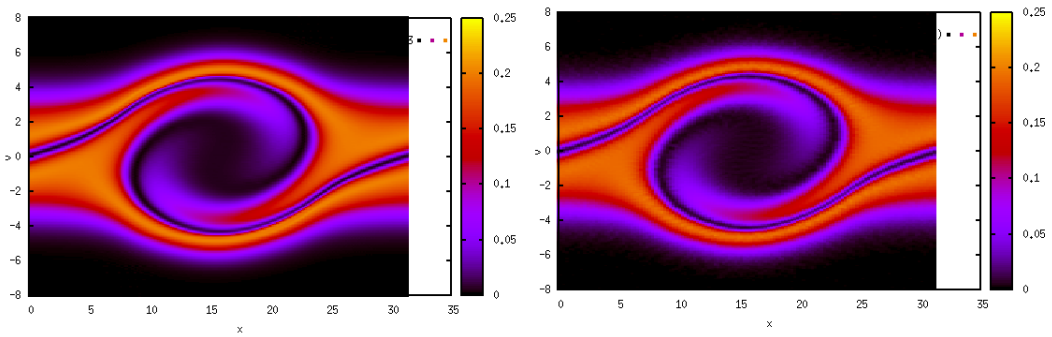

Figure 2: The distribution function of the two-stream test case at time $t = 25$. The first coordinate is the space coordinate x_1. The second coordinate is the velocity v_1. Left: reduced Vlasov-Poisson method. Right: the PIC method.

5.2. Two-stream instability

In this example we test the validity of the reduced Vlasov model in a simplified one-dimensional situation. The initial distribution function represents two Gaussian beams of charged particles moving in opposites directions. The distribution is a stationary solution of the Vlasov-Maxwell system. We add a small perturbation in order to trigger instability.

$$f_0(x,v) = (1 + \epsilon cos(kx_1)) \frac{1}{2\sqrt{2\pi}} \left(e^{-\frac{(v_1-v_0)^2}{2}} + e^{-\frac{(v_1+v_0)^2}{2}} \right).$$

The value of parameters for this test case are $k = 0.2$, $\epsilon = 5 \times 10^{-3}$ and $v_0 = 3$. In the one-dimensional Maxwell equations, we consider a very large speed of light, which leads in practice to the resolution of the Poisson equation.

We compare our reduced Vlasov-Poisson method to a well-validated Particle-In-Cell (PIC) scheme [6]. The distribution function is plotted at time $t = 25$ and $t = 50$ in Figures 2. We observe an excellent agreement between the two approaches [4, 7].

5.3. Reduced Vlasov-Maxwell

In this section, we present preliminary Vlasov-Maxwell numerical results obtained with the reduced approach in two dimensions. We consider a cloud of charged particles in the center of a square domain Ω. Because the particles repel each other, the cloud expands with time. We plot the evolution of the total charge in the domain at different times. We also represent the distribution function in the velocity space at a given point $(x_1, x_2) = (0.37, 0.5)$. The initial distribution function is

$$f(x_1, x_2, v_1, v_2, 0) = -q(x_1)q(x_2)q(v_1)q(v_2),$$

where the smooth function q has its support in $[-1/2, 1/2]$ and satisfies $\int_{r=-\infty}^{\infty} q(r)dr = 1$. On the boundary of Ω, we apply homogeneous Silver-Müller conditions. We use a mesh of Ω with $8 \times 8 = 64$ cells. The velocity mesh is of order $d = 2$. It contains 305 Gauss-Lobatto nodes. We plot the charge evolution on Figure 3.

It would be also possible to plot other quantities, such as the EM field or the dependance of the distribution in the velocity space

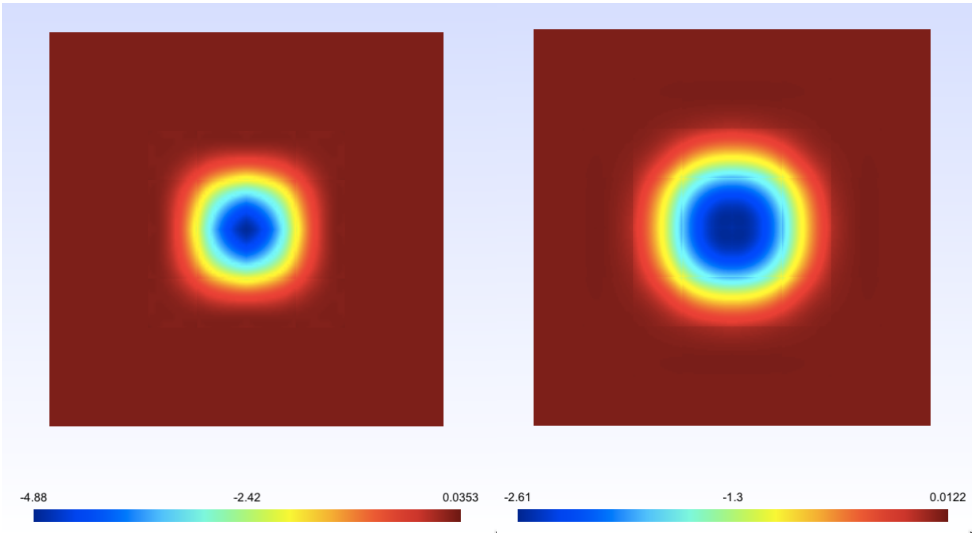

Figure 3: evolution of the charge in the computational domain, $t = 0$ (left), $t = 0.5$ (right).

6. CONCLUSION

We have presented CLAC, an electromagnetic simulation tool. This tool is intended for industrial practical computations. It is accelerated by two parallelism technologies: MPI and OpenCL, which allows to run CLAC on GPU clusters.

The design of the software is generic, in such a way that it can potentially solve any non-linear hyperbolic system of conservation laws. We have then proposed a reduction technique that allows to rewriting the six-dimensional Vlasov equation as a three-dimensional hyperbolic system. We have presented preliminary one-dimensional and two-dimensional numerical results obtained with this reduction approach.

ACKNOWLEDGMENT

This work has benefited from several supports: from the french defense agency DGA, from the Labex ANR-11-LABX-0055-IRMIA and from the AxesSim company.

REFERENCES

1. F. Bourdel, P. A. Mazet, and P. Helluy, "Resolution of the non-stationary or harmonic Maxwell equations by a discontinuous finite element method. Application to an E.M.I. (electromagnetic impulse) case", *Proceedings of the 10th international conference on computing methods in applied sciences and engineering*, pp. 405–422, 1992.
2. G. Cohen, X. Ferrieres, and S. Pernet, "A spatial high-order hexahedral discontinuous Galerkin method to solve Maxwell's equations in time domain", *Journal of Computational Physics*, vol. 217, no. 2, pp. 340–363, 2006.
3. A. Klöckner, T. Warburton, J. Bridge, and J. S. Hesthaven, "Nodal discontinuous Galerkin methods on graphics processors", *J. Comput. Phys.*, vol. 228, no. 21, pp. 7863–7882, 2009.
4. P. Helluy, N. Pham, and A. Crestetto, "Space-only hyperbolic approximation of the Vlasov equation", *ESAIM: Proceedings*, vol. 43, pp. 17–36, 2013.
5. P. Helluy, L. Navoret, N. Pham, and A. Crestetto, "Reduced Vlasov-Maxwell simulations", http://hal.archives-ouvertes.fr/hal-00957045, March 2014.
6. C.K. Birdsall and A.B. Langdon, *Plasma Physics via Computer Simulation*, Institute of Physics (IOP), Series in Plasma Physics, 1991.
7. P. Helluy, N. Pham, and L. Navoret, "Hyperbolic approximation of the Fourier transformed Vlasov equation", http://hal.archives-ouvertes.fr/hal-00872972, 2014.